REMARKS

Claims 1-17 and 19 are pending in this application. By this Amendment, the specification and claims 1-9, 15 and 16 are amended to correct minor informalities, such as to correct antecedence inconsistencies unrelated to patentability. No new matter is added and the claims are not narrowed by these amendments.

Applicants appreciate the courtesies extended to Applicants' representative by Examiner Pham during the July 22, 2005 telephone interview.

Reconsideration of the application is respectfully requested.

I. Amendment Entry After Final Rejection

Entry of this amendment is proper under 37 CFR §1.116 because the amendments: a) place the application in condition for allowance for all the reasons discussed herein; b) do not raise any new issues requiring further search or consideration; c) place the application in better condition for appeal if necessary; and d) address formal requirements of the Final Rejection and preceding Office Action. Accordingly, Applicants respectfully request entry of this Amendment.

II. The Claims Satisfy the Requirements under 35 U.S.C. §112, second paragraph

The Final Office Action rejects claims 1-16 and 19 under 35 U.S.C. §112, second paragraph, as being indefinite. This rejection is respectfully traversed. Independent claims 1, 15 and 16 do not recite "such as" language, but instead further limit the claim to define the first type as comprising one of the recited list. The language of the independent claims is a proper recitation of a *Markush* group. See MPEP §2173.05(h). Withdrawal of the rejection under 35 U.S.C. §112, second paragraph, is respectfully requested.

III. Claims 1-17 and 19 Define Patentable Subject Matter

The Final Office Action rejects claims 1-3, 8-10, 12-16 and 19 under 35 U.S.C. §103(a) over U.S. Patent No. 6,636,972 to Ptacek et al. ("Ptacek") in view of U.S. Patent No.

5,892,951 to Safonov; rejects claims 4 and 5 under 35 U.S.C. §103(a) over Ptacek in view of Safonov and further in view of U.S. Patent No. 5,276,854 to Court et al. ("Court"); rejects claims 6 and 7 under 35 U.S.C. §103(a) over Ptacek in view of Safonov and further in view of U.S. Patent No. 5,613,117 to Davidson et al. ("Davidson"); and rejects claim 11 under 35 U.S.C. §103(a) over Ptacek in view of Safonov and further in view of "Java decompilers compared" by Dyer (*Java World*, July 1997). These rejections are respectfully traversed.

Ptacek and Safonov, alone or in combination do not teach or suggest a method of operating a computer system for evaluating a programming language statement that includes a first and a second sub-statement including evaluating the first sub-statement as a first evaluation to determine a first evaluation success result if the first evaluation succeeds or else a distinguished value if the first evaluation fails; the distinguished value being a value not included in a first range of possible evaluation success results of the first sub-statement, determining whether to evaluate the second sub-statement as a second evaluation, and if so, evaluating the second sub-statement to determine a second evaluation success result if the second evaluation succeeds or else the distinguished value if the second evaluation fails; a second range of possible evaluation success results of the second sub-statement not including the distinguished value, and determining an evaluation result of the statement depending on at least whether the first evaluation of the first sub-statement succeeds or fails, wherein the first evaluation success result of the first sub-statement and the second evaluation success result of the second sub-statement belong to a first type in a set of types and the distinguished value belongs to a second type not in the set of types, each type in the set of types is not a supertype of the second type, and the first type in the set of types comprises one from the set of integer, float, Boolean, sequence, tuple, structure, multi-set, dictionary, string and enumeration, as recited in claim 1, and similarly recited for an article of manufacture for use in a computer system operating the above method, as recited in claim 15.

Ptacek and Safonov also do not teach or suggest a system for evaluating a programming language statement and determining an evaluation result of the statement including a memory for storing the statement that includes a first and a second sub-statement, a processor for determining the evaluation result of the statement; the evaluation result of the statement depending on whether a sub-evaluation of the first and second sub-statements succeeds or fails; the processor being capable of evaluating the first sub-statement and determining a first evaluation success result if the sub-evaluation succeeds, or a distinguished value if evaluation fails; the processor being capable of evaluating the second sub-statement and determining a second evaluation success result if the sub-evaluation succeeds, or the distinguished value if the sub-evaluation fails; the first evaluation success result of the first sub-statement and the second evaluation success result of the second sub-statement belong to a first type in a set of types and the distinguished value belongs to a second type not in the set of types; wherein each type in the set of types is not a supertype of the second type; and the first type in the set of types comprises one from the set of integer, float, Boolean, sequence, tuple, structure, multi-set, dictionary, string and enumeration, as recited in claim 16.

Ptacek discloses a method for building a script to perform a network security audit. In particular, Ptacek teaches a computer software system 220 directing a computer system 100 on a network. The system 220 includes a Custom Attack Simulation Language (CASL) 270 to test networks including a source code description or Script 271 compiled on a CASL compiler 273. Also, Ptacek teaches CASL programs composed of statements of control constructs and expressions, including loops and conditionals, with expressions assigned to variables (e.g., 0, 1) equivalent to logical results (e.g., “false”, “true”) (col. 5, lines 15-61, col. 9, lines 53-64, col. 10, lines 41-47 and Figs. 2A and 2B of Ptacek).

Safonov discloses a method for compiling source code that verifies selected semantic attributes while building a parse tree, instead of afterwards. In particular, Safonov teaches constructing parse sub-trees of the construct "E" at step 306 that call attribute semantic routines, and then constructing a tree node for E at step 308 (col. 5, lines 30-61 and Fig. 3 of Safonov).

Applicants respectfully assert that the Final Office Action does not explain how Ptacek and Safonov allegedly teach evaluating a first sub-statement, determining whether to evaluate the second sub-statement, and if so, evaluating the second sub-statement to determine a second evaluation success result if the second evaluation succeeds or else the distinguished value if the second evaluation fails, and determining an evaluation result of a statement depending on at least whether the first sub-statement succeeds or fails, with first and second evaluation success results belonging to a first type in a set of types, and a distinguished value belonging to a second type not in the set of types, as recited in claims 1 and 15.

The Final Office Action fails to show how Ptacek allegedly provides for the conditional evaluation sequence as recited in claims 1 and 15, but instead addresses generalized functionalities associated with Ptacek that, as admitted in the Final Office Action, do not disclose features related to the distinguished value belonging to a type not in the set of types. Moreover, Applicants assert that Safonov merely teaches tree node construction with pre-construction verification, rather than distinguishing types for values. In addition, neither Ptacek nor Safonov teaches or suggests that each type in the set of types is not a supertype of the second type, and that the first type in the set of types comprises one from the set of integer, float, Boolean, sequence, tuple, structure, multi-set, dictionary, string and enumeration, as recited in the independent claims. The Final Office Action appears not to have accorded proper patentable weight to the recited type characteristics, as the Final Office Action does not indicate any disclosure in Ptacek or Safonov allegedly corresponding to these features.

Court, Davidson and Dyer do not compensate for the deficiencies of Ptacek and Safonov outlined above for claim 1. Court does not teach, disclose or suggest the additional features recited in claims 4 and 5. Davidson also does not teach, disclose or suggest the additional features recited in claims 6 and 7. Dyer also does not teach, disclose or suggest the additional features recited in claim 11.

Court discloses software simulation of multiple CPUs having a common design specified in Boolean logic. In particular, Court teaches a method for packing program variables with data from multiple CPUs wherein the variables can be used in program steps to simultaneously evaluate the Boolean logic for a plurality of CPUs (col. 3, lines 41-70 of Court). However, the Final Office Action does not explain how Court allegedly teaches concurrently evaluating first and second sub-statements with the evaluation result of the statement being distinguished value if one or both of the first and second evaluations of the sub-statements fail.

Davidson discloses an optimizing compiler. In particular, Davidson teaches a compiler framework 10 including a front end 20, a shell 11 and a code generator 12. The front end 20 can be configured for a specific coding language, while the shell 11 can be tailored to operate under an interchangeable host operating system 13 (col. 5, lines 38-61 and Fig. 1 of Davidson). However, the Final Office Action does not explain how Davidson allegedly teaches evaluating a second sub-statement depending on the success of evaluating the first sub-statement with the evaluation result of the statement being distinguished value if one or both of the first and second evaluations of the sub-statements fail.

Dyer discloses a comparison of Java decompilers: DejaVu, Mocha and WingDis, for several error categories. Regarding the portion cited in the Final Office Action, Dyer teaches a list comparison source code reconstruction by the DejaVu decompiler based on integer variable `CTL_PC` (page 9 of Dyer). However, the Final Office Action does not explain how such unannotated source code allegedly provides teaching of an unordered action system to one of ordinary skill in the art.

Further, there is no motivation to combine features related to the security audit script building method of Ptacek with the semantically verified parse tree constructing compiler of Safonov, the multiple CPU simulation of Court, the interchangeable compilers and operating systems of Davidson and the Java decompiler comparison of Dyer. The Final Office Action has not established a proper motivation for a *prima facie* case of obviousness.

In particular, asserts at page 8 that one of ordinary skill would have motivation to incorporate the teaching of Safonov into the teaching of Ptacek to include a distinguished value belonging to a second type not in the set of types in order to more efficiently identify TYPE mismatching or semantic errors, thereby minimizing computation time. The Final Office Action similarly asserts at pages 11 and 12 the reduction of execution time as motivation to incorporate into the teachings of Ptacek the teachings for concurrent evaluation or for an unordered action system of first and second sub-statements of Court or Dyer, respectively. The Final Office Action also asserts at page 12 that the motivation for incorporating the teachings of Davidson into that of Ptacek would be to yield real mode values. Applicants respectfully disagree and submit that a generic objective of reducing computation time is not sufficient to establish motivation. There must be some expectation of success that the combined teachings will achieve these alleged advantages.

Safonov teaches verifying selected semantic attributes for compiling source code, whereas Ptacek teaches programs with expressions equivalent to logical results, representing a process that an artisan of ordinary skill would interpret as being short-circuited and thereby compromising its operation by verifying and thereby fixing attributes prematurely. Also, the teachings of multiple-CPU simulation of Court and the optimizing compiler of Davidson appear to show minimal association with performing a network security audit taught by

Ptacek. Similarly, the alternate forms of interpretations of compiled code in Dyer would not be applicable to one of ordinary skill in the art to modify the logic structures of Ptacek, particularly in light of the non-intuitive syntax used by the DejaVu decompiler.

A *prima facie* case of obviousness for a §103 rejection requires satisfaction of three basic criteria: there must be some suggestion or motivation either in the references or knowledge generally available to modify the references or combine reference teachings, a reasonable expectation of success, and the references must teach or suggest all the claim limitations (MPEP §706.02(j)). Applicants assert that the Final Office Action fails to satisfy these requirements with Ptacek, Safonov, Court, Davidson and Dyer.

For at least these reasons, Applicants respectfully assert that the independent claims 1, 15 and 16 are patentable over the applied references. The dependent claims are likewise patentable over the applied references for at least the reasons discussed, as well as for the additional features they recite. Consequently, all the claims are in condition for allowance. Thus, Applicants respectfully request Applicant respectfully requests that the rejections under 35 U.S.C. §103 be withdrawn.

IV. Conclusion

In view of the foregoing, Applicants respectfully submit that this application is in condition for allowance. Favorable reconsideration and prompt allowance are earnestly solicited.

Should the Examiner believe that anything further is desirable in order to place this application in even better condition for allowance, the Examiner is invited to contact Applicants' undersigned representative at the telephone number listed below.

Respectfully submitted,

James A. Oliff
Registration No. 27,075

Gerhard W. Thielman
Registration No. 43,186

JAO:GWT/gwt

Date: July 26, 2005

OLIFF & BERRIDGE, PLC
P.O. Box 19928
Alexandria, Virginia 22320
Telephone: (703) 836-6400

DEPOSIT ACCOUNT USE
AUTHORIZATION
Please grant any extension
necessary for entry;
Charge any fee due to our
Deposit Account No. 24-0037